

## PATENT ABSTRACTS OF JAPAN

(11)Publication number : 03-164835  
 (43)Date of publication of application : 16.07.1991

51)Int.Cl. G06F 9/45

21)Application number : 01-302149

(71)Applicant : HITACHI LTD  
 HITACHI TOHOKU SOFTWARE KK

22)Date of filing : 22.11.1989

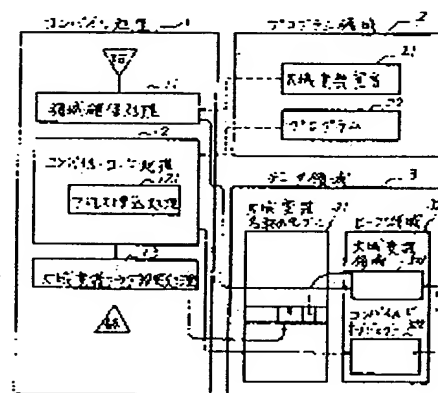
(72)Inventor : HIRAIDE TOSHIMICHI  
 HIROSE TADASHI  
 HARA MIYUKI  
 TOMONAGA KATSUKO

#### 54) COMPILING METHOD FOR GLOBAL VARIABLE PROCESSING IN INTERPRETER TYPE LANGUAGE PROCESSING SYSTEM

##### 57)Abstract:

**PURPOSE:** To increase the execution speed of a compiled object by performing the processing, which interprets a specification statement to reserve a variable area of a global variable in accordance with the type and the size of the variable, and the processing, which sets the address of the reserved variable area to program areas where the global variable is referred and updated, at the time of area reserving processing.

**CONSTITUTION:** An area reserving processing 11, an address setting processing 121 in a compile load processing 12, and a global variable flag setting processing 13 are provided. The area reserving processing 11 reserves a global variable area in accordance with a global variable declaration 21 included in a program area 2 and stores its address. A compiled object 322 where the address of a processing object (the address in a global variable area 321) is reserved for the global variable in a program 22 is generated based on the stored address. Thereafter, a change inhibition flag of the global variable name as the object in global variable name table 31 is set to inhibition by the global variable flag setting processing 13. Thus, the execution speed of the compiled object of the program is increased.



#### LEGAL STATUS

Date of request for examination]

Date of sending the examiner's decision of rejection]

Kind of final disposal of application other than the examiner's decision of rejection or application converted registration]

Date of final disposal for application]

Patent number]

Date of registration]

Number of appeal against examiner's decision of rejection]

Date of requesting appeal against examiner's decision of rejection]

Date of extinction of right]

Copyright (C); 1998,2003 Japan Patent Office

## ⑫ 公開特許公報(A)

平3-164835

⑤ Int. Cl.<sup>5</sup>

識別記号

庁内整理番号

⑬ 公開 平成3年(1991)7月16日

G 06 F 9/45

8724-5B

G 06 F 9/44

3 2 2 F

審査請求 未請求 請求項の数 3 (全7頁)

⑭ 発明の名称 インタプリタ型言語処理系における大域変数処理のコンパイル方法

⑮ 特 願 平1-302149

⑯ 出 願 平1(1989)11月22日

⑰ 発 明 者 平 出 利 道 宮城県仙台市青葉区一番町2丁目4番1号 日立東北ソフトウェア株式会社内

⑰ 発 明 者 広 瀬 正 神奈川県川崎市麻生区王禅寺1099番地 株式会社日立製作所システム開発研究所内

⑰ 発 明 者 原 幸 神奈川県横浜市戸塚区戸塚町5030番地 株式会社日立製作所ソフトウェア工場内

⑰ 出 願 人 株式会社日立製作所 東京都千代田区神田駿河台4丁目6番地

⑰ 出 願 人 日立東北ソフトウェア株式会社 宮城県仙台市青葉区一番町2丁目4番1号

⑰ 代 理 人 弁理士 小川 勝男 外1名  
最終頁に続く

## 明 細 書

## 1. 発明の名称

インタプリタ型言語処理系における大域変数処理のコンパイル方法

## 2. 特許請求の範囲

1. 大域変数処理機能を持つインタプリタ型言語処理系において、予めコンパイル対象となるソースプログラムと共に定義された大域変数の名称と変数の型と変数の大きさに従って大域変数の領域を静的に確保する処理(領域確保処理)をし、大域変数を参照・更新するプログラム領域に領域確保処理で確保した領域のアドレスを埋め込む処理(アドレス埋込処理)を行ないながらコンパイル対象プログラムも機械語形式のプログラム(コンパイルドオブジェクト)に変換し、データ領域に書き込み処理(コンパイル・ロード処理)を行なうことを特徴としたインタプリタ型言語処理系における大域変数処理のコンパイル方法。

2. 請求項1記載の方法において、大域変数名称

のテーブルに変更不可フラグ領域を設定し、コンパイル・ロード処理後にアドレス埋込処理を行なった大域変数名称の変更不可フラグを不可に設定し、コンパイル対象外のプログラムから呼び出された大域変数の変数領域の生成・削除処理あるいは変数領域の参照・更新処理の発生時に変更不可フラグを判定し、不可ならばその生成・削除処理あるいは参照・更新処理を異常終了し、不可以外ならばその処理を行うことを特徴とするインタプリタ型言語処理系における大域変数処理のコンパイル方法。

3. 請求項1記載の方法において、大域変数領域の領域確保処理時に大域変数名称のテーブルの変更不可フラグを判定し、不可でなくかつ確保する領域と同一の大きさを持つ領域が既存するならば領域確保処理を行わずに既存の領域を再利用することを特徴とするインタプリタ型言語処理系における大域変数処理のコンパイル方法。

3. 発明の詳細な説明

## 〔産業上の利用分野〕

本発明は、インタプリタ型言語処理系の処理方法に関する。

## 〔従来の技術〕

例えばインタプリタ型言語処理系の1つであるPROLOG処理系には、例えば、「日立クリエイティブワークステーション2050、プロログ（株式会社日立製作所ソフトウェア工場、昭和63年6月発行）の第87頁」に論じられているように大域変数（グローバル変数）が用意されている。しかしながら、この種の大域変数はそのものが動的に生成・削除されること、デバグ終了部分から順次プログラムを部分的にコンパイルすることができるインタプリタ型言語開発環境の特徴を確保するため、従来はコンパイル対象プログラムで大域変数を参照・更新を機械語形式のプログラムに変換していなかった。従って大域変数の参照・更新処理は、実行時に大域変数の変数領域のアドレスを大域変数名称のテーブルから取り出していた。

た大域変数名称のテーブルから変数領域のアドレスを取り出す処理を不要とし、コンパイルドオブジェクトの実行速度の高速化を実現する。

また、大域変数名称のテーブルに変更不可フラグ領域を設定し、アドレス埋込処理を行なった大域変数名の変更不可フラグを不可に設定し、コンパイル対象外のプログラムからの大域変数の変数領域の生成・削除処理あるいは参照・更新処理に対して変更不可フラグを参照し、不可ならばその処理を異常終了し、さもないければその処理を実行する。

さらに、領域確保処理時に宣言文による大域変数の名称から変更不可フラグを参照し、不可以外でかつ確保する領域と同一の大きさを持つ領域が存在するならば既存の領域を再利用する。

## 〔作用〕

大域変数の領域確保処理時には、大域変数の宣言文中の変数の大きさから確保する変数領域の大きさと確保した変数領域のアドレスを求めることができ、コンパイル対象プログラムで大域変数領

## 〔発明が解決しようとする課題〕

上記従来技術は、コンパイル対象プログラム内に大域変数を参照・更新するプログラム部分があるとそこはインタプリティブに実行せざるをえず、コンパイルドオブジェクト全体の実行速度に多大な影響を与えるもので、コンパイルドオブジェクトの実行速度の高速化を妨げているという問題があった。

本発明の目的は、大域変数処理を含むプログラムのコンパイルドオブジェクトの実行速度の高速化を図ることにある。

## 〔課題を解決するための手段〕

上記目的を達成するために、本発明においては、大域変数の名称と変数の型と変数の大きさを記述する宣言文を用意し、領域確保処理時にその宣言文を解釈して変数の型と変数の大きさから大域変数の変数領域を確保する処理と、大域変数を参照・更新するプログラム領域に上述で確保した変数領域のアドレスを埋め込む処理をすることで、従来コンパイルドオブジェクトの実行時に行っていた

域を参照・更新する部分は、参照・更新するアドレスが決定できた機械語形式のプログラムに変換することができる。それによつて、従来コンパイルドオブジェクトの実行時に大域変数名称のテーブルから変数領域のアドレスを取り出し、参照・更新するアドレスを決定していた処理が不要となり、コンパイルドオブジェクトの実行速度の高速化が実現できる。

また、大域変数名称のテーブルに変更不可フラグ領域を設定し、コンパイル対象となつた大域変数の変更不可フラグ領域のフラグを不可に設定しておき、さらにコンパイル対象外のプログラムからの大域変数領域の生成・削除処理あるいは参照・更新処理に対して変更不可フラグ領域のフラグを判定することにより、フラグが不可ならばその処理が異常終了でき、さもないければその処理が実行できる。したがって、コンパイル対象プログラムを実行プログラムの一部に限定した場合でも、正しくない動作の発生を防止できる。なぜなら、コンパイル対象プログラム内の大域変数に対して

確保した変数領域を、コンパイル対象外のプログラムからの生成・削除処理あるいは参照・更新処理で、破壊される場合は、上記異常終了を発生させ、その旨をプログラム作成者に知らせることができるからである。

さらに、前述の宣言文による大域変数の名称と変数領域の大きさに基づき、コンパイル対象外のプログラムの当該大域変数にたいする生成・削除処理時で、大域変数名称のテーブルの変更不可フラグを処理実行前に判定し、もし変更不可フラグが設定されているとき、領域確保処理時には新規に変数領域を確保することせず、また領域削除処理時には領域を削除しないことにより、コンパイル対象プログラムを実行プログラムの一部分に限定しても従来コンパイルした部分と同等の動作を保證することができる。

#### 〔実施例〕

第1図は、本発明による大域変数処理を含むプログラムのコンパイル処理の概要を表すフローチャートである。図において点線は情報の流れを表

を生成する。

その後、大域変数フラグ設定処理13で大域変数名称のテーブル31の対象となつた大域変数名称の変更不可フラグを不可に設定する。

第2図は、第1図の大域変数名称のテーブルの詳細と大域変数名称のテーブル31と大域変数領域321の関係を示した図である。この図を用いて領域確保処理11と大域変数フラグ設定処理13を詳述する。

大域変数名称のテーブル31の1エントリは以下の5つのフィールド、すなわち大域変数名称311と変更不可フラグ領域313と大域変数領域321のアドレス領域314と他の領域312、315とからなる。312、315は、本発明と直接関係ないので説明を省略する。

領域確保処理11では、大域変数の宣言（大域変数名称、変数の型、変数の大きさ）の変数の型と変数の大きさに基づきプログラム領域2の中に変数領域を確保し、そのアドレスを該当する大域変数に対応する1つのエントリの大域変数名称の

し、実線は処理の流れを表す。

コンパイル処理とは、プログラム領域2内に格納されるプログラム22をコンパイル・ロード処理12で解析・変換処理を行つた結果をデータ領域3内のヒープ領域32にコンパイルドオブジェクト322として生成するものである。従つて、従来のデータ領域3には大域変数名称のテーブル31とヒープ領域に含まれるコンパイルドオブジェクト322が存在する。

本発明では、従来のコンパイル処理に領域確保処理11とコンパイル・ロード処理12内のアドレス埋込処理121と大域変数フラグ設定処理13を設けたことに特徴がある。第1図を用いて各処理の概要を説明する。

領域確保処理11はプログラム領域2内に含まれる大域変数宣言21から大域変数領域を確保し、そのアドレスを格納する。プログラム22中にある大域変数は、前述で格納したアドレスから処理対象のアドレス（大域変数領域321内のアドレス）を確定したコンパイルドオブジェクト322

アドレス領域314に格納する。

大域変数フラグ設定処理13は、変数領域321を確保した大域変数名称の変更不可フラグ領域312を不可とする処理である。

第3図は、第1図のコンパイルドオブジェクト322と大域変数領域321との関係を示した図である。この図を用いてアドレス埋込処理121を詳述する。

大域変数処理221を含むプログラム22のコンパイル・ロード処理12において、大域変数領域321のアドレスは既に領域確保処理11で変数の確保をしたときに決定されており、大域変数名称のテーブルのアドレス領域314に格納されている。そこで、アドレス埋込処理121は、大域変数処理221の対象アドレスを求めコンパイルドオブジェクト322（機械語形式のプログラム）に組み入れる。

第4、5図は、前述のアドレス埋込処理121で生成される機械語形式のプログラムへの変換例を示したものである。

第4図は、大域変数領域の更新処理プログラム411を含むプログラム41が、アドレス埋込処理121により機械語形式のプログラム42内の命令421に変換される例を示したものである。

global\_nameは大域変数の名称を示し、Indexは変数領域内の対象となるフィールドを指定するインデックスを示し、\_は変数領域内の対象フィールドの値(非更新値)を判定しないことを示し、1は更新値を示す。すなわち、更新処理プログラム例41のglobal\_update(global\_name, Index, \_1)411は、大域変数名称global\_nameのIndex番目のフィールドの値を整数値の1に変更する命令である。この処理が機械語形式のプログラム42では、MOVE #1, global\_name\_addr+Index 421に変換できる。ここでglobal\_name\_addrは、領域確保処理11で確保した変数領域のアドレスを示し、global\_name\_addr+Indexは変数領域内の対象フィールドのアドレスを示す。すなわちこの変換処理において、アドレス埋込処理121は、領域確保処理11で確保した変数領域のアドレス(global\_name\_addr+Index)を大域変数名称のテーブル31のアドレス領域314から求めることが出来るため、変数領域内の処理対象フィールドのアドレス(global\_name\_addr+Index)を求め、機械語形式のプログラムの命令521~522に変換する。

し、global\_name\_addr+Indexは変数領域内の対象フィールドのアドレスを示し、FAILは判定(参照)が失敗したときに実行するアドレスを示すが、本発明と直接関係ないため詳細の説明は省略する。すなわちこの変換処理において、アドレス埋込処理121は、領域確保処理11で確保した変数領域のアドレス(global\_name\_addr)を大域変数名称のテーブル31のアドレス領域314から求めることが出来るため、変数領域内の処理対象フィールドのアドレス(global\_name\_addr+Index)を求め、機械語形式のプログラムの命令521~522に変換する。

第6図は、変更不可フラグを用いた大域変数の生成・削除処理フローを示している。

変数の大きさが正の時に、大域変数の生成処理61を行う。大域変数名称のテーブルの対応する変数名称の1フィールド内の変更不可フラグ領域321のフラグを判定しフラグが不可、つまり変数領域が使用中の場合は生成処理を異常終了611する。さもなければ、変数領域が未使用とみなし

name\_addr)を大域変数名称のテーブル31のアドレス領域314から求めることが出来るため、変数領域内の処理対象フィールドのアドレス(global\_name\_addr+Index)を求め、機械語形式のプログラムの命令421に変換する。

第5図は、大域変数領域の参照処理プログラム511を含むプログラム51が、アドレス埋込処理121により機械語形式のプログラム52内の命令521に変換される例を示したものである。

global\_nameは大域変数の名称を示し、Indexは変数領域内の対象となるフィールドを指定するインデックスを示し、1は参照値を示す。すなわち、参照処理プログラム例51のglobal\_unify(global\_name, Index, 1) 511は、大域変数名称global\_nameのIndex番目のフィールドの値が整数値の1かどうか判定(参照)する命令である。この処理が機械語形式のプログラム52では、COMP #1, global\_name\_addr+Index 521, BNE FAIL 522に変換できる。ここでglobal\_name\_addrは、領域確保処理11で確保した変数領域のアドレスを示

変更不可フラグ領域321のフラグを不可にする処理612を行う。

また、変数の大きさが正以外の場合は大域変数の削除処理62を行う。削除処理62は、大域変数名称のテーブルの対応する変数名称の1フィールド内の変更不可フラグ領域321のフラグを判定し、フラグが不可、つまり変数領域が使用中の場合は、確保した変数領域はそのままにし変更不可フラグ領域321のフラグを不可以外にする621処理を行う。さもなければ変数領域が未使用とみなし、削除処理62を異常終了622する。

第7図は、コンパイル対象プログラム内の大域変数と非コンパイル対象プログラム内の大域変数の処理を示した図である。

コンパイル対象プログラム71は、大域変数の宣言711、712と大域変数生成処理713、714と大域変数の更新処理715、716を含むプログラムである。非コンパイル対象プログラム72は、大域変数生成処理721、大域変数更新処理722、大域変数参照処理723を含むプ

ログラムである。

ここで、コンパイル対象プログラムをコンパイル処理するとデータ領域73には、大域変数名称のテーブル731内のフィールド7311に大域変数名a.b.cを、同内のフィールド7312に大域変数名c.d.eを登録し、a.b.cの変数領域732とc.d.eの変数領域733を確保し、a.b.cの変更不可フラグ領域7313のフラグとc.d.eの変更不可フラグ領域7314のフラグを不可に設定する。

この図は、非コンパイル対象プログラム72をインタプリティブに実行させると、大域変数の生成、更新、参照の各処理は、変数名称c.d.eの変更不可フラグ領域7314のフラグを判定し、不可であることからコンパイル対象プログラム71により変数領域733が使用中であると判断し、異常終了することを示している。

第8図は、変更不可フラグを用いた領域確保処理11の詳細フローである。

宣言文（大域変数名称、変数の型、変数の大き

さ）の解析処理81を行ない変数の型と変数の大きさに基づき、確保する変数領域の大きさを求める処理82をし、領域の確保を行う。このとき、大域変数名称のテーブルのフィールドの変更不可フラグ領域のフラグが不可以外でかつ確保する変数領域と大きさが等しい領域が存在する場合に、既存の変数領域が再利用可能となり、再利用処理83を行なう。再利用処理83では、大域変数名称のテーブル内の対象フィールドの変数名称を変更する処理831を行なう。さもなければ、新たに変数領域を確保する処理（新規変数領域の確保処理84）を行なう。新規変数領域の確保処理84では、変数領域の確保処理841をし、大域変数名称のテーブル内に新たなフィールドを設定し、そのフィールド内に大域変数名称を登録する処理（大域変数名称の登録処理842）をし、処理841で得たアドレスを同フィールド内のアドレス格納領域に格納する処理（変数領域アドレス格納処理843）を行なう。

〔発明の効果〕

本発明によれば、インタプリタ型言語処理系における大域変数処理方法において、コンパイルドオブジェクト生成時に大域変数の処理対象アドレスの決定が可能となる。

また、デバツク終了部分から順次、プログラムを部分的にコンパイルすることができるインタプリタ型言語のプログラム開発環境の特徴を従来のまま確保できる。

この結果、従来、コンパイルドオブジェクトの実行時に大域変数の処理対象アドレスを求めていたことに比較してより高速実行可能となる。

#### 4. 図面の簡単な説明

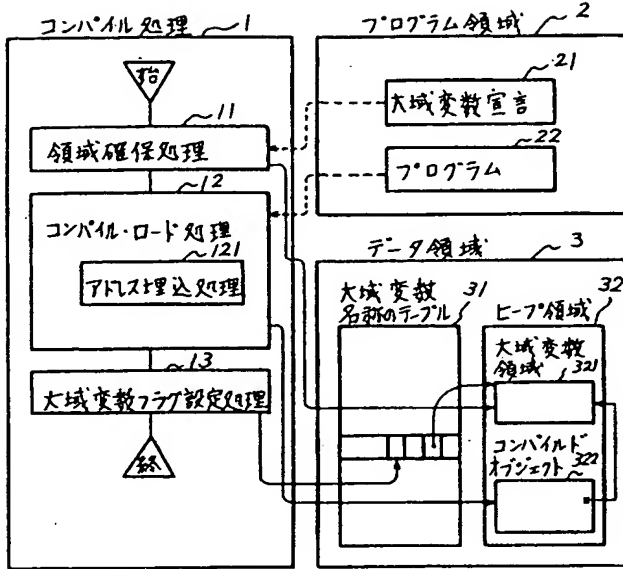
第1図は本発明の概要を示すコンパイル処理フロー図、第2、3図は本発明の一実施例の処理の詳細説明図、第4図は本発明の実施例の大域変数更新処理の機械語形式プログラムへの変換を示す説明図、第5図は本発明の実施例の大域変数参照処理の機械語形式プログラムへの変換を示す説明図、第6図は本発明の実施例の大域変数生成、削除処理フロー図、第7図は本発明の実施例のコン

パイル対象プログラムと非コンパイル対象プログラム内の大域変数処理の関連の説明図、第8図は本発明の実施例の大域変数領域の確保処理フロー図である。

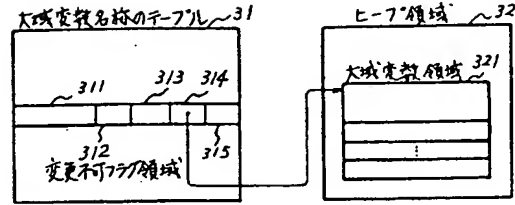
代理人 弁理士 小川勝男



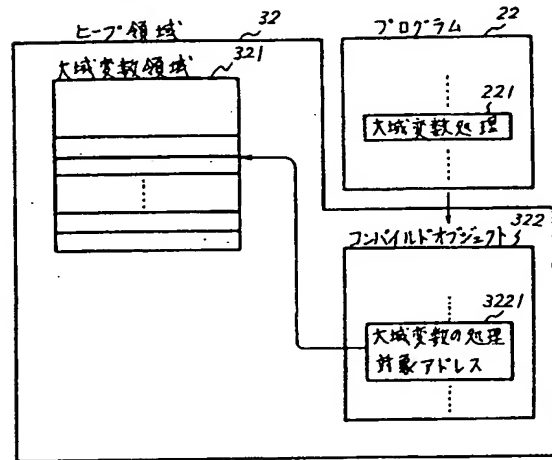
第 1 図



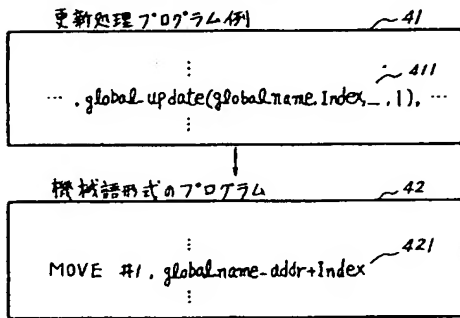
第 2 図



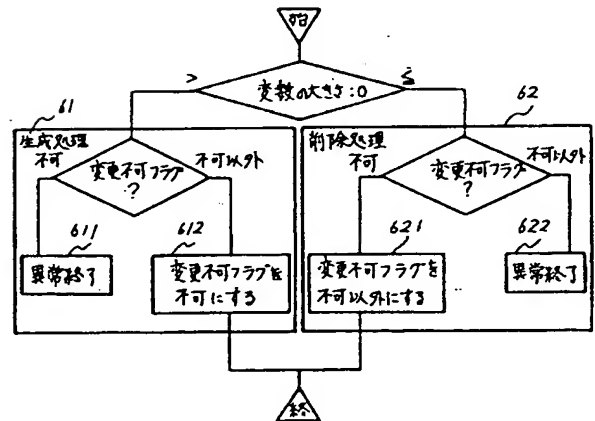
第 3 図



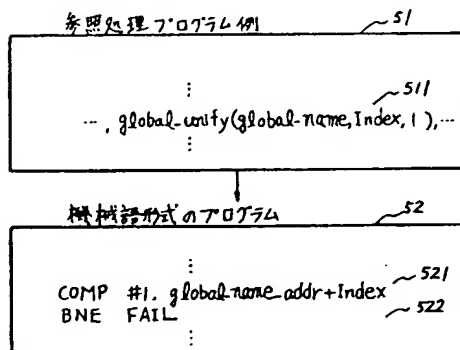
第 4 図



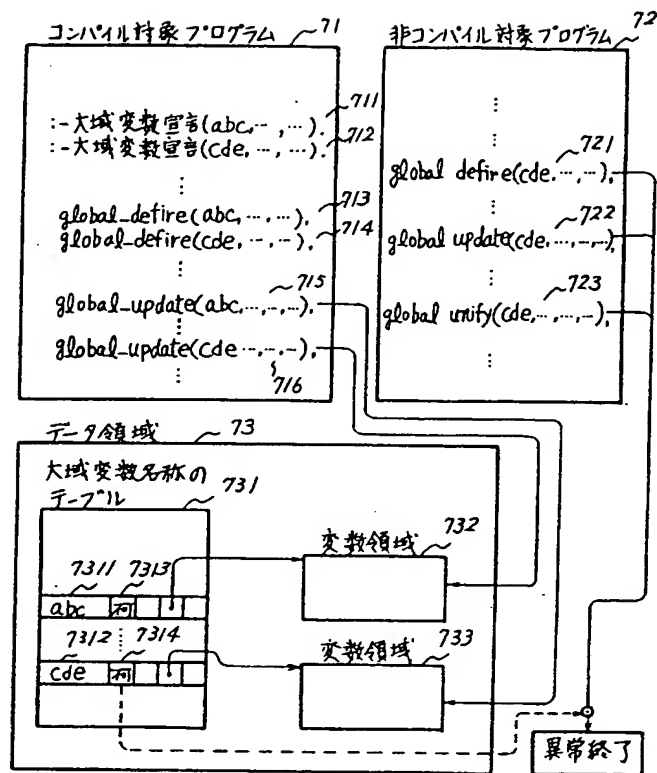
第 5 図



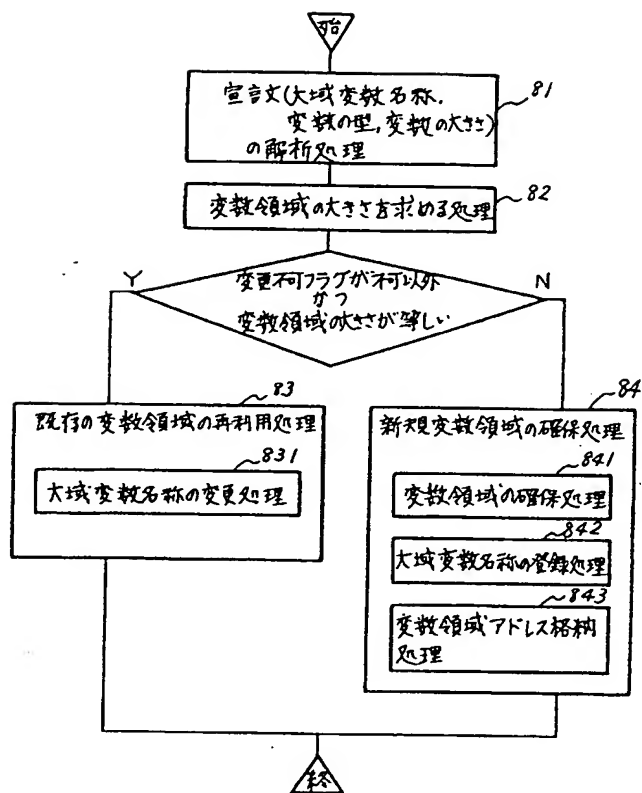
第 6 図



第 7 図



第 8 図



第 1 頁の続き

⑦発 明 者 友 永 佳 津 子 神奈川県横浜市戸塚区戸塚町5030番地 株式会社日立製作所ソフトウェア工場内